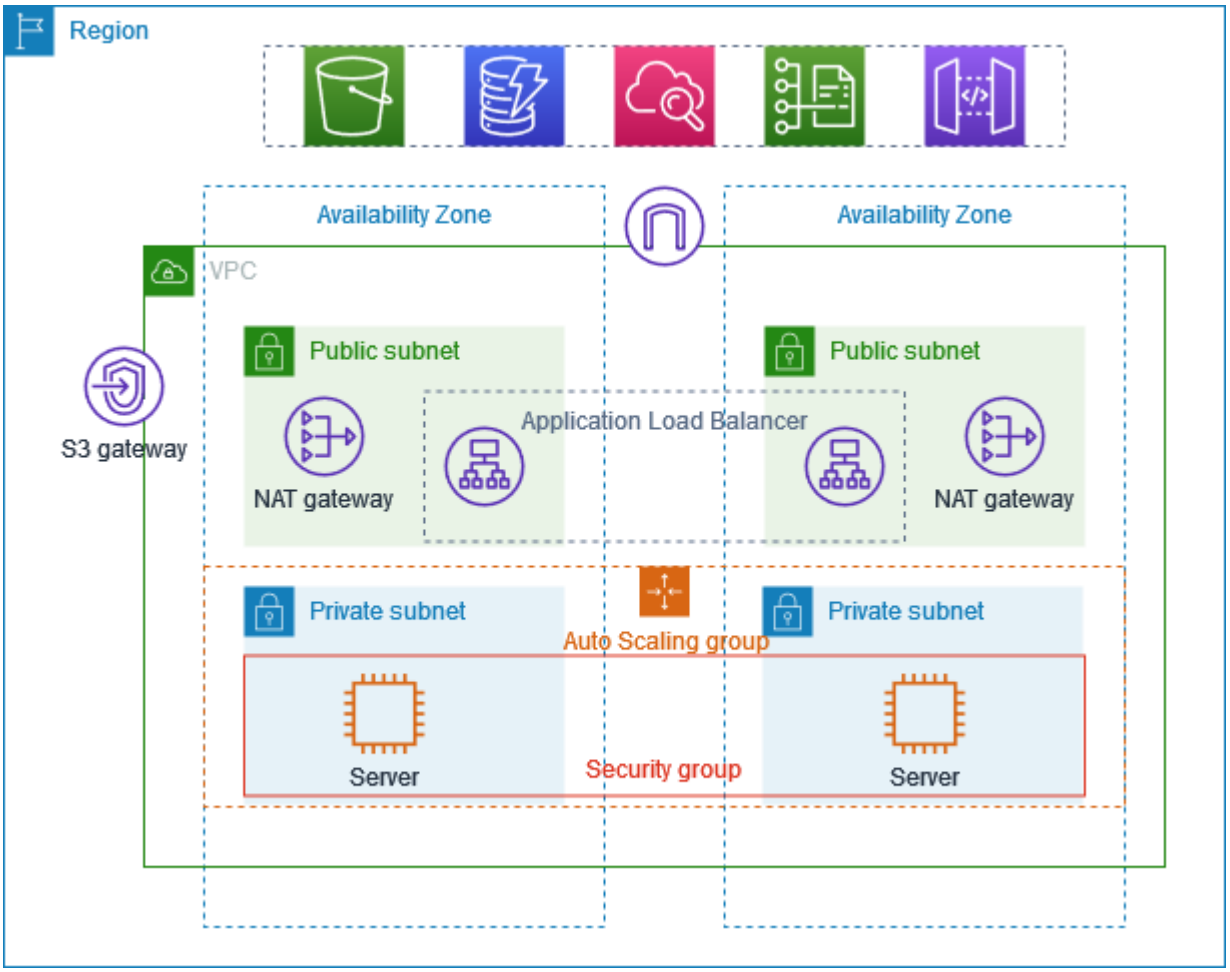
AWS –Production Deployement Infrastructure

Overview

The following diagram provides an overview of the resources included in this example. The VPC has public subnets and private subnets in two Availability Zones. Each public subnet contains a NAT gateway and a load balancer node. The servers run in the private subnets, are launched and terminated by using an Auto Scaling group, and receive traffic from the load balancer. The servers can connect to the internet by using the NAT gateway. The servers can connect to Amazon S3 by using a gateway VPC endpoint.



1.VPC Creation

To create the VPC

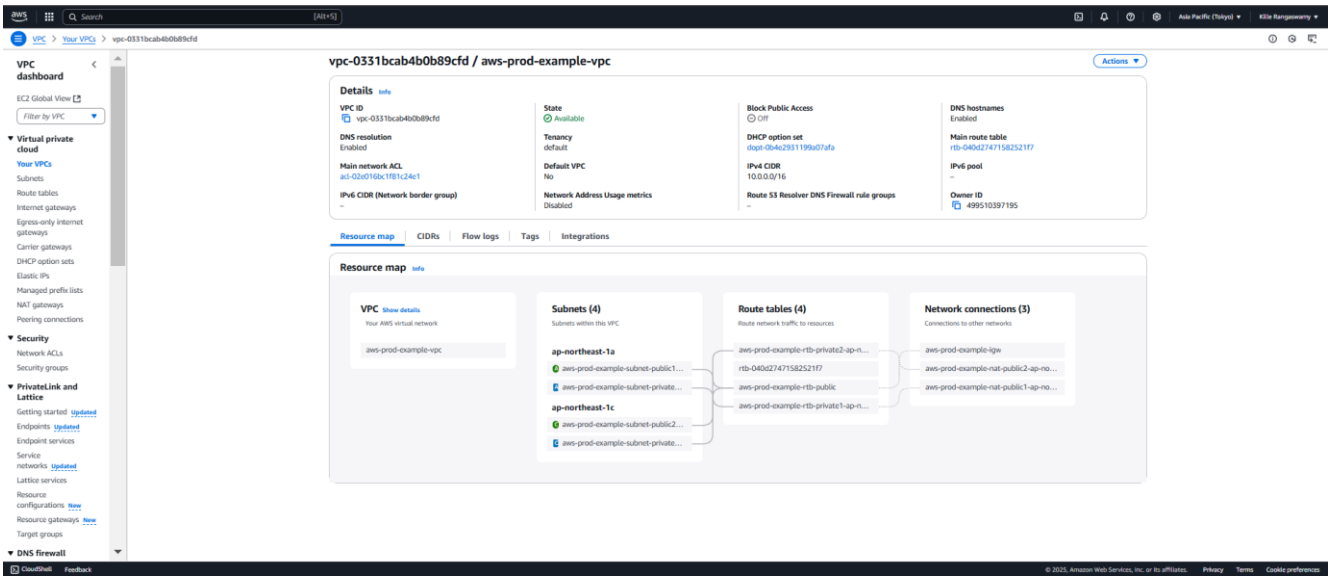
1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. On the dashboard, choose **Create VPC**.
3. For **Resources to create**, choose **VPC and more**.

Configure the VPC:

1. For **Name tag auto-generation**, enter aws-prod-example.
2. For **IPv4 CIDR block**, you can keep the default suggestion, or alternatively you can enter the CIDR block required by your application or network.
3. If your application communicates by using IPv6 addresses, choose **IPv6 CIDR block**, Amazon-provided IPv6 CIDR block.

Configure the subnets:

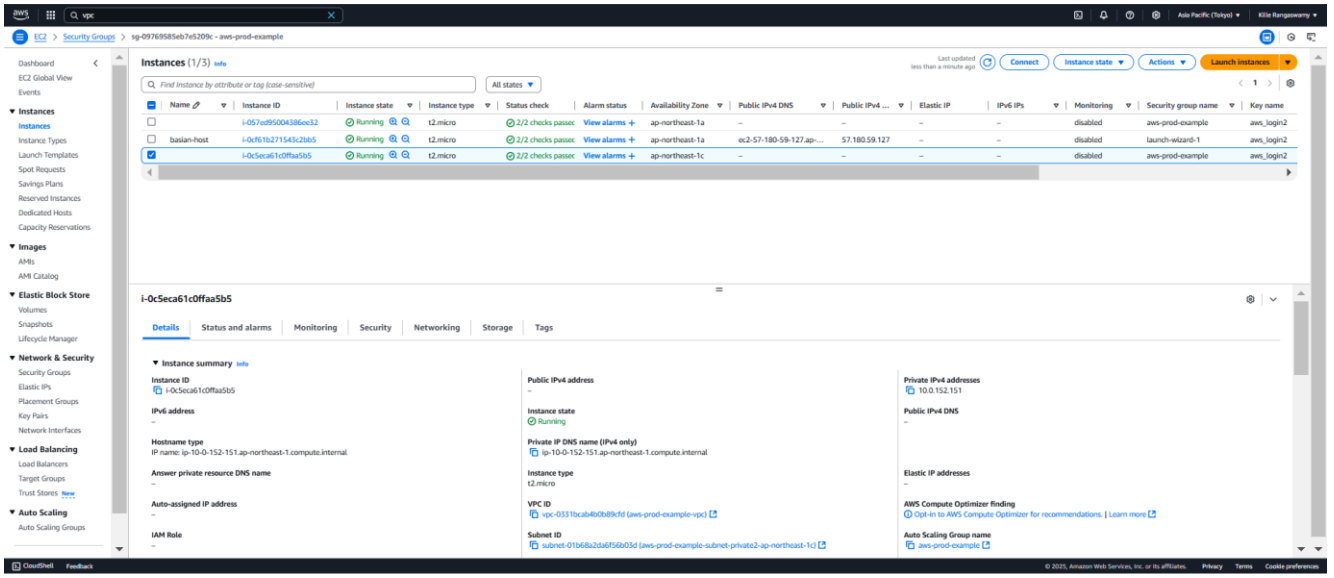
1. For **Number of Availability Zones**, choose 2, so that you can launch instances in multiple Availability Zones to improve resiliency.
2. For **Number of public subnets**, choose 2.
3. For **Number of private subnets**, choose 2.
4. You can keep the default CIDR block for the public subnet, or alternatively you can expand **Customize subnet CIDR blocks** and enter a CIDR block. For more information, see **Subnet CIDR blocks**.
5. For **NAT gateways**, choose 1 per AZ to improve resiliency.
6. If your application communicates by using IPv6 addresses, for **Egress only internet gateway**, choose Yes. Otherwise No IPv6
7. For **VPC endpoints**, if your instances must access an S3 bucket, keep the S3 Gateway default. Otherwise, instances in your private subnet can't access Amazon S3. There is no cost for this option, so you can keep the default if you might use an S3 bucket in the future. If you choose **None**, you can always add a gateway VPC endpoint later on.OR **otherwise select None**
8. For **DNS options**, clear **Enable DNS hostnames**.
9. Choose **Create VPC**.



2.Auto scaling groups creation

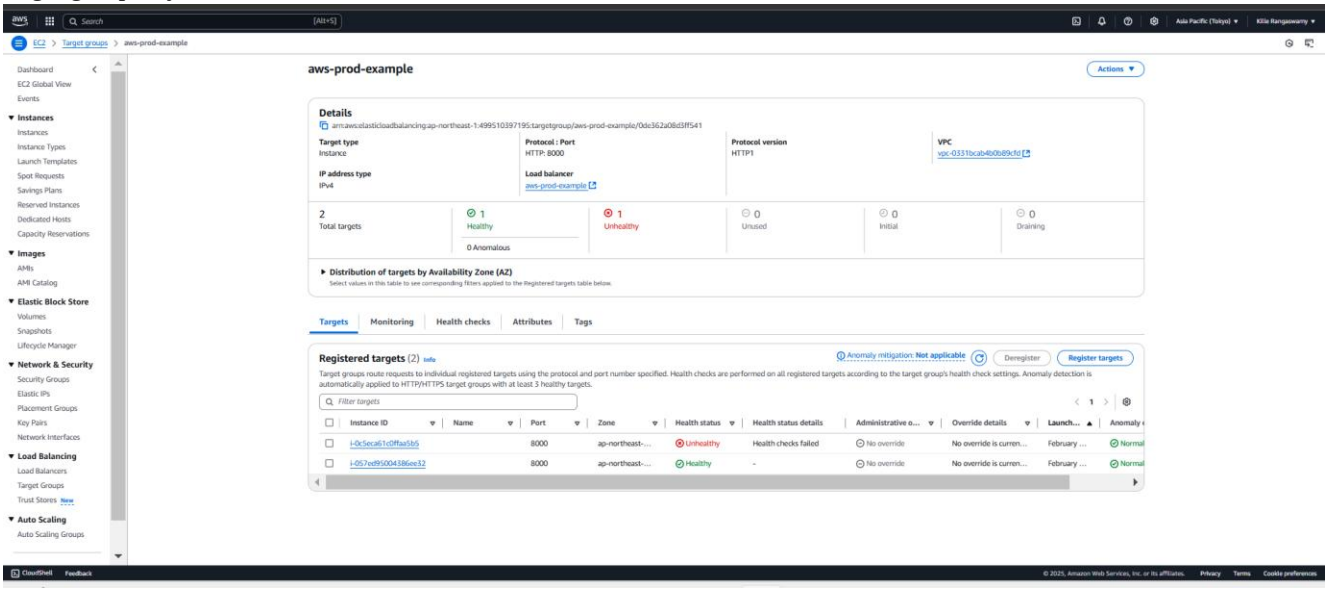
To launch instances by using an Auto Scaling group:

1. Create a launch template to specify the configuration information needed to launch your EC2 instances by using Amazon EC2 Auto Scaling.
2. Create an Auto Scaling group, which is a collection of EC2 instances with a minimum, maximum, and desired size.



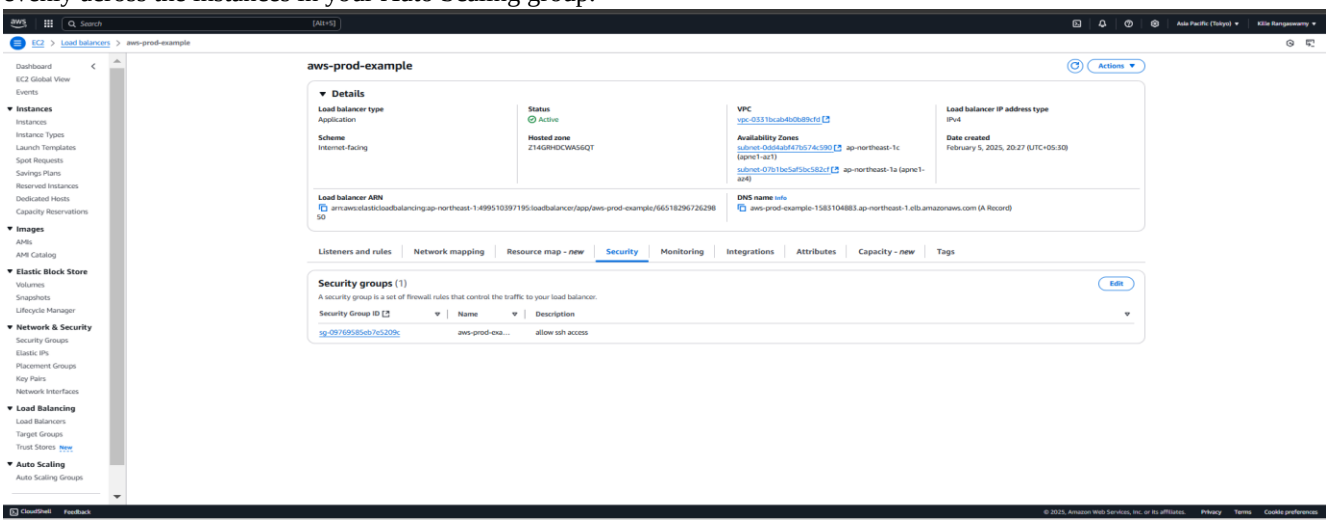
3. Target Groups creation

- Create a **Target Group**, which is a set of EC2 instances that the load balancer will distribute traffic to, and attach the target group to your load balancer



3.Load Balancer creation

Create a **Load Balancer** (for example, an Application Load Balancer or Network Load Balancer) to distribute traffic evenly across the instances in your Auto Scaling group.



4. Deploying Applications

Connect your instances from your local system through the terminal where you have downloaded your Pem file and fonfigure the with the below commands to connect and deploye Applications in both private instances

```
PS C:\Users\MI\Downloads> scp -i aws_login2.pem aws_login.pem ubuntu@57.180.59.127:/home/ubuntu
C:\WINDOWS\System32\OpenSSH\scp.exe: stat local "aws_login.pem": No such file or directory
PS C:\Users\MI\Downloads> scp -i aws_login2.pem aws_login2.pem ubuntu@57.180.59.127:/home/ubuntu
aws_login2.pem                100% 1674     3.8KB/s   00:00
PS C:\Users\MI\Downloads> ssh -i aws_login2.pem ubuntu@57.180.59.127
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1021-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Wed Feb  5 14:07:20 UTC 2025

  System load:  0.0               Processes:             109
  Usage of /:   25.1% of 6.71GB   Users logged in:       0
  Memory usage: 21%               IPv4 address for enX0: 10.0.6.41
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
```

```
ubuntu@ip-10-0-6-41:~$ ls
aws_login2.pem
```

```
ubuntu@ip-10-0-6-41:~$ ssh -i aws_login2.pem ubuntu@10.0.141.30
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1021-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Wed Feb  5 14:22:48 UTC 2025

  System load:  0.0               Processes:             105
  Usage of /:   25.1% of 6.71GB   Users logged in:       0
  Memory usage: 20%               IPv4 address for enX0: 10.0.141.30
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
```

```
ubuntu@ip-10-0-141-30:~$ python3
Python 3.12.3 (main, Nov  6 2024, 18:32:19) [GCC 13.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
KeyboardInterrupt
>>>
KeyboardInterrupt
>>> 10.0.141.30
[1]+  Stopped                 python3
ubuntu@ip-10-0-141-30:~$ vim index.html
```

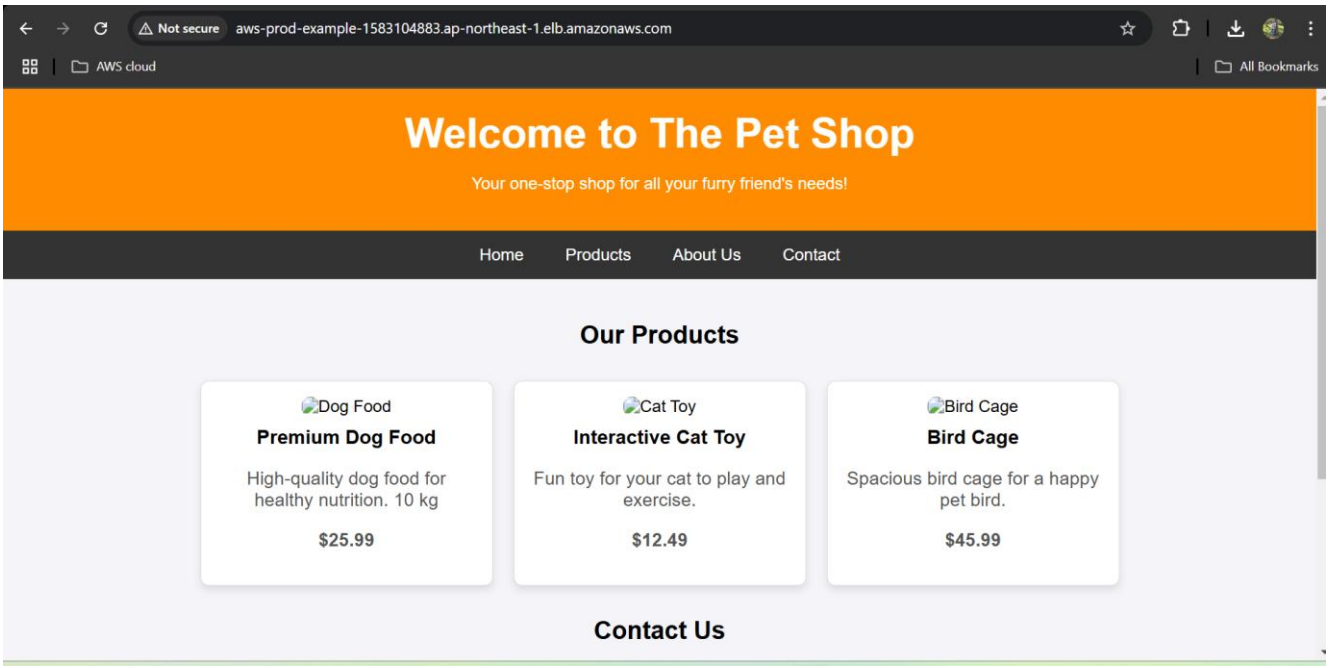
```
ubuntu@ip-10-0-141-30:~$ vim index.html
ubuntu@ip-10-0-141-30:~$ ls
index.html
```

```
ubuntu@ip-10-0-141-30:~$ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.0.7.126 - - [05/Feb/2025 16:18:43] "GET / HTTP/1.1" 200 -
10.0.7.126 - - [05/Feb/2025 16:18:58] "GET / HTTP/1.1" 200 -
10.0.21.81 - - [05/Feb/2025 16:19:03] "GET / HTTP/1.1" 200 -
10.0.7.126 - - [05/Feb/2025 16:19:28] "GET / HTTP/1.1" 200 -
10.0.21.81 - - [05/Feb/2025 16:19:33] "GET / HTTP/1.1" 200 -
10.0.7.126 - - [05/Feb/2025 16:19:58] "GET / HTTP/1.1" 200 -
10.0.21.81 - - [05/Feb/2025 16:20:03] "GET / HTTP/1.1" 200 -
10.0.7.126 - - [05/Feb/2025 16:20:28] "GET / HTTP/1.1" 200 -
10.0.21.81 - - [05/Feb/2025 16:20:33] "GET / HTTP/1.1" 200 -
10.0.7.126 - - [05/Feb/2025 16:20:58] "GET / HTTP/1.1" 200 -
10.0.21.81 - - [05/Feb/2025 16:21:03] "GET / HTTP/1.1" 200 -
10.0.7.126 - - [05/Feb/2025 16:21:28] "GET / HTTP/1.1" 200 -
```

5.Outputs

Deployed 2 different Demo static application in AWS production Environment servers

server-1



Server-2